Troubleshooting Cisco ASA firewall via HTTPs, sem telnet, ssh ou ASDM

Existem diversas formas de se administrar um Cisco firewall:

Algumas delas:

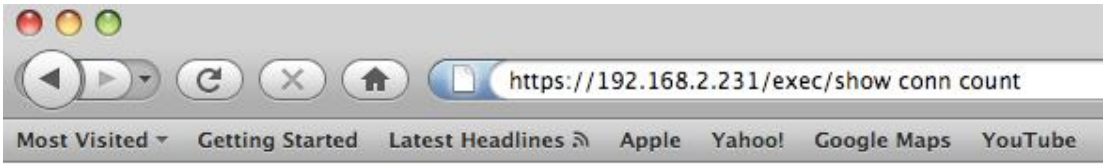
- via telnet (tcp 23)
- via ssh (tcp 22)
- via https (ASDM tcp 443)

Uma forma alternativa de se executar comandos no ASA seria via HTTPs diretamente, sem o uso do Cisco ASDM (Adaptive Security Device Manager). Usando a seguinte URL:

https://<ip_do_firewall>/exec/comando

Alguns exemplos de comandos usandos durante a resolução de problemas de performance em um firewall Cisco:

show conn count: Mostra o número máximo e o número atual de conexões através do firewall.

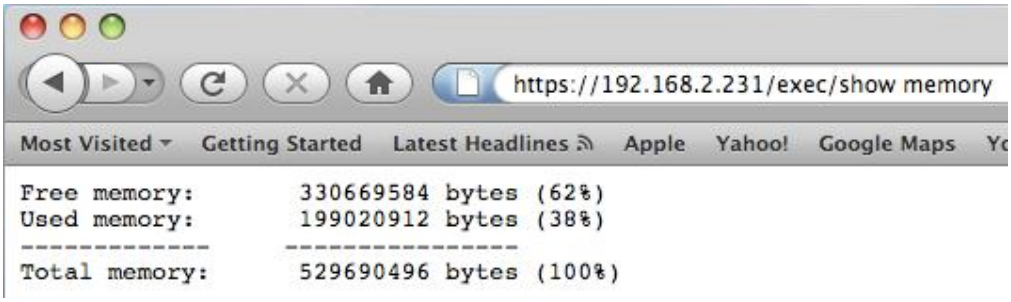



```
204 in use, 3996 most used
```

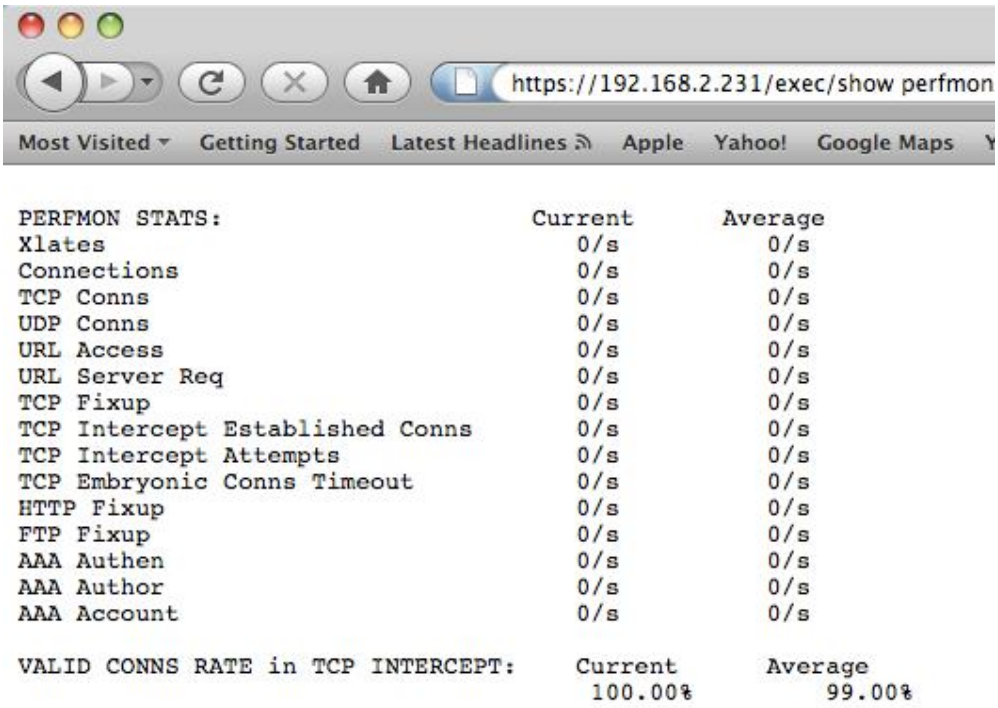
show traffic: Mostra a quantidade de tráfego passando pelo firewall em um determinado período de tempo.

```
zumInternet:
        received (in 1265855.538 secs):
                587157448 packets       371711608566 bytes
                2 pkts/sec      293003 bytes/sec
        transmitted (in 1265855.538 secs):
                530286999 packets       132748942568 bytes
                1 pkts/sec      104000 bytes/sec
      1 minute input rate 1031 pkts/sec,  1319933 bytes/sec
      1 minute output rate 592 pkts/sec,  42687 bytes/sec
      1 minute drop rate, 0 pkts/sec
      5 minute input rate 1111 pkts/sec,  1309680 bytes/sec
      5 minute output rate 676 pkts/sec,  52713 bytes/sec
      5 minute drop rate, 0 pkts/sec
Cables:
        received (in 1265855.538 secs):
                470993679 packets       117210212522 bytes
                2 pkts/sec      92003 bytes/sec
        transmitted (in 1265855.538 secs):
                527020735 packets       364395590525 bytes
                2 pkts/sec      287003 bytes/sec
      1 minute input rate 565 pkts/sec,  36844 bytes/sec
      1 minute output rate 1002 pkts/sec,  1314076 bytes/sec
      1 minute drop rate, 1 pkts/sec
      5 minute input rate 795 pkts/sec,  53239 bytes/sec
      5 minute output rate 1229 pkts/sec,  1323410 bytes/sec
      5 minute drop rate, 1 pkts/sec
```

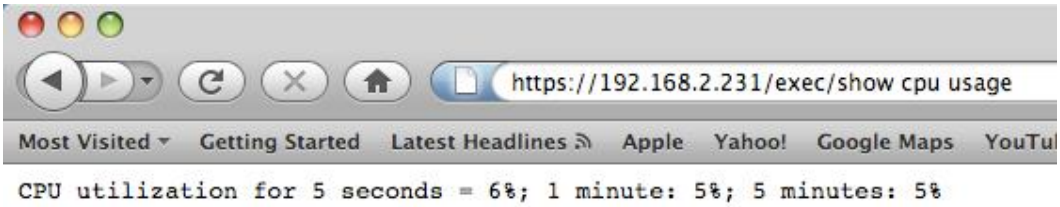
show memory: Verifica a quantidade de memória do sistema (total, livre e usada).



show perfmon: Verifica a quantidade e tipo de tráfego passando pelo firewall



show cpu usage : verifica o uso da CPU do firewall



Outros comandos úteis durante o troubleshooting de problemas de performance são:

show blocks
show xlate
show interface
show processes

Protegendo ataques ao BGP em routers Cisco - parte 1

BGP (Border Gateway Protocol) é um protocolo de roteamento usualmente usado em provedores de serviço (ISP).

Empresas pequenas muitas vezes utilizam apenas rotas estáticas para se conectarem a um único provedor e usando apenas uma conexão. E, por isto, estão livres dos potencias problemas que podem

enfrentar ao utilizarem um protocolo de roteamento como o BGP nos roteadores de borda com a INTERNET.

Existem diversos tipos de ataques que o utilizam o BGP para interromper o serviço em uma rede.

Normalmente o conhecimento e técnicas para se evitar este tipo de ataque está concentrado em engenheiros/técnicos dentro dos Provedores.

Somando a isso, o fato de que o número de empresas utilizando BGP em seus roteadores vem aumentando, resolvi postar aqui algumas dessas técnicas que podem (e deveriam) ser utilizadas por qualquer engenheiro de rede.

Dois tipos de ataques são:

- **Manipulação de rotas** – Nesse caso um equipamento altera o conteúdo da tabela de roteamento BGP do roteador local.
- **Negação de Serviço (DoS)** – Nesse caso o ataque é feito aos recursos do roteador (cpu/memória/tabela de rotas/tabela BGP). A intenção é esgotar estes recurso para impedir o funcionamento do roteador

Proteção 1: Protegendo os peers (vizinhos) com uma senha (MD5).

A primeira e essencial técnica (e mais conhecida) é utilizar senha entre os roteadores rodando BGP. Isto evita que um roteador ou qualquer maquina simulando BGP forme um peer com o seu roteador e divulgue rotas que podem interromper o trafego da sua rede. Isto é evita o ataque manipulação de rotas. Não é possível ler a senha transmitida capturando os pacotes na rede, uma vez que ela é enviada em forma de um "hash" MD5.

O comando usado para isso é:

neighbor {*ip-address* | *peer-group-name*} **password** *string*

http://www.cisco.com/en/US/docs/ios/12_2/iproute/command/reference/1rfbgp1.html#wp1260412

Proteção 2: Confirmando o TTL (tempo de vida) dos pacotes BGP

Essa técnica em contraste com a anterior é bem desconhecida dos engenheiros de redes.

Por padrão, em uma conexão eBGP (BGP externo - entre dois AS (Sistemas Autônomos) diferentes), os pacotes enviados entre os roteadores vizinhos tem um TTL = 1.

Isto se dá, porque normalmente seções como essa se dão entre roteadores diretamente conectados (e as vezes entre as interfaces loopbacks desse roteadores - neste caso usa-se o comando ***neighbor ebgp-multihop*** para aumentar o TTL dos pacotes).

Um tipo ataque utilizando uma "chuva" de pacotes forjados destinados ao roteador (TCP 179) irá consumir os recursos de CPU do mesmo. Nesse caso, a Proteção 1 não impede o ataque e pode até contribuir para o aumento do consumo de CPU. Esse tipo de ataque pode ser desferido de qualquer lugar da INTERNET.

Uma técnica para se evitar esse ataque consiste em se verificar o tempo de vida dos pacotes BGP enviados. Uma vez que é muito dificil utilizar de forma eficiente pacotes com TTL forjados.

O comando utilizado para isso é:

neighbor *ip-address ttl-security hops hop-count*

Onde *hop-count* indica a distância em saltos para o vizinho BGP.

A verificação feita é: O TTL do pacote recebido é maior que 255 - hopcount ?

Caso verdadeiro o pacote é aceito, caso contrário é descartado e nenhum ICMP é enviado.

Nota: Esse comando substitui o comando neighbor ebgp-multihop. E estes são mutuamente exclusivos.

Um exemplo básico da utilização desse comando:

```
router bgp 12345
no synchronization
neighbor 200.1.1.1 remote-as 19000
neighbor 200.1.1.1 ttl-security hops 2
no auto-summary
```

*Pacotes enviados a este roteador com TTL menor que 253 (255 - 2) serão descartados.

Podemos verificar isto com o comando **show ip bgp neighbor ip**

Router# **show ip bgp neighbors | i External BGP neigh**
External BGP neighbor may be up to 2 hops away

***traduzindo:** Vizinho Externo BGP pode estar até 2 hops de "distância"

Essa técnica não protege contra:

- Ataques desferidos de dentro da própria rede (ou rede diretamente conectada ao roteador);
- Ataques entre peers iBGP (não existe TTL limite);
- Vizinhos que estão a vários saltos de distância (utilizando ebgp-multihop)

Como fazer meu CISCO ASA/PIX aparecer em um traceroute?

Usualmente, costumamos usar todos os meios para tornarmos os firewalls o mais transparentes possível em uma rede, por questões óbvias de segurança.

Entretanto, de tempo em tempo recebemos perguntas como essa:
"Como fazer meu CISCO ASA/PIX aparecer em um traceroute?"

A configuração padrão de um firewall CISCO (ASA/PIX) não decrementa o TTL dos pacotes L3 que trafegam pelo mesmo. Mesmo se estes estão configurados como um "hop" extra na rede. Isto é como um roteador.

Para tanto, é necessária seguinte configuração:

*Nota: PIX/ASA versão de software acima de 7.x

policy-map global_policy
class class-default
set connection decrement-ttl

É necessário, também, a criação de uma access-list (ACL) permitindo o retorno de pacotes ICMP tipo 11 (time-exceded) e tipo 3 código 3 (unreachables)

access-list OUTSIDE_IN extended permit icmp any any time-exceeded
access-list OUTSIDE_IN extended permit icmp any any unreachable

access-group OUTSIDE_IN in interface outside

Antes do mudança:

r1#traceroute 136.1.22.22

Type escape sequence to abort.
Tracing the route to 136.1.22.22

1 136.1.122.2 356 msec 296 msec 96 msec
2 136.1.22.22 192 msec * 248 msec

Após a mudança:

r1#traceroute 136.1.22.22

Type escape sequence to abort.
Tracing the route to 136.1.22.22

1 136.1.122.12 36 msec 100 msec 96 msec
2 136.1.122.2 72 msec 132 msec 208 msec
3 136.1.22.22 124 msec * 80 msec

Filtrando tráfego intra-vlan (camada 2) - vlan access-maps - CISCO VACL

O uso comum de um ACL consiste na aplicação de filtros em uma porta de camada 3. Esse filtro irá apenas filtrar pacotes que estão cruzando a interface de camada 3 onde a ACL foi aplicada.

Como máquinas em uma mesma Vlan podem se comunicar diretamente na camada 2 (MAC para MAC), não é possível evitar a comunicação intra-vlan com esse tipo de filtro. Para isso utilizamos as VACLs.

Em um switch camada 3 (layer 3) temos 3 tipos de ACLs:

Routed ACL: São aplicadas as interfaces com IP (routed interfaces ou SVI (switched vlan intefaces: interface vlans) e servem para filtrar pacotes que "cruzam" por esta interface. Podem ser aplicadas em qualquer direção (filtram pacotes entrando ou saindo da interface: (inbound or outbound)).

Port ACL: São ACLs (IP ou MAC) usadas para filtrar tráfego em uma porta de camada 2 (layer 2 - switchport mode access ou mode trunk) vindos das máquinas conectadas a esta porta (filtra somente tráfego que entra na interface: inbound).
*Uma Port ACL tem precedência sobre os outros tipos de ACL.

VACLs ou Vlan access-maps: São utilizadas para filtrar pacotes enviados dentro de uma mesma VLAN. Essas Vlans podem filtrar tanto IP como outros tipos de protocolos usando-se as Ethernet (MAC) ACLs. Este tipo de ACL apenas controla tráfego no switch onde foram configuradas.

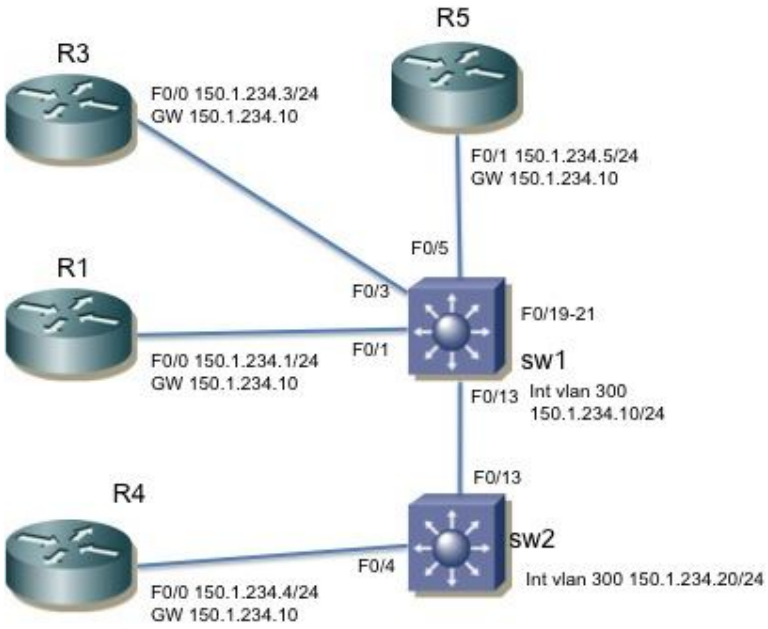


Figura 1

Antes de aplicar qualquer filtro (ACL) na topologia acima, vemos que temos conectividade entre as máquinas na VLAN 300 e entre essas máquinas e IPs fora desta VLAN (através do gateway 150.1.234.40).

```
R1#ping 150.1.234.255 rep 1

Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 150.1.234.255, timeout is 2 seconds:

Reply to request 0 from 150.1.234.10, 1 ms
Reply to request 0 from 150.1.234.3, 1 ms
Reply to request 0 from 150.1.234.5, 1 ms
Reply to request 0 from 150.1.234.4, 1 ms
Reply to request 0 from 150.1.234.40, 1 ms
Reply to request 0 from 150.1.234.30, 1 ms
R1#p 150.1.200.40

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.200.40, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/4 ms

R1#sh arp
Protocol Address Age (min) Hardware Addr Type Interface
Internet 150.1.234.5 6 001b.0cfa.9f69 ARPA FastEthernet0/0
Internet 150.1.234.4 6 001b.2a55.ff70 ARPA FastEthernet0/0
Internet 150.1.234.1 - 001b.0cfa.9eb0 ARPA FastEthernet0/0
Internet 150.1.234.3 7 001b.2a55.e338 ARPA FastEthernet0/0

R3#ping 255.255.255.255 rep 1

Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 255.255.255.255, timeout is 2 seconds:

Reply to request 0 from 150.1.234.10, 4 ms
Reply to request 0 from 150.1.234.5, 4 ms
Reply to request 0 from 150.1.234.4, 4 ms
Reply to request 0 from 150.1.234.1, 4 ms
Reply to request 0 from 150.1.234.40, 4 ms
Reply to request 0 from 150.1.234.20, 4 ms
```

Usando uma **Port ACL**, podemos filtrar pacotes IP vindos do R1 (150.1.234.1).

Neste exemplo, uma ACL que filtra os pacotes destinados ao roteador R4 (150.1.234.4) vindos do R1 foi aplicada a interface f0/1 do sw1. Lembrando que uma ACL deste tipo somente pode ser aplicada com Inbound.

```
sw1(config)#access-list 101 deny ip any host 150.1.234.4
sw1(config)#access-list 101 permit ip any any
sw1(config)#int f0/1
sw1(config-if)#ip access-group 101 out
^
% Invalid input detected at '^' marker.

sw1(config-if)#ip access-group 101 in
```

Agora, R1 não pode mais se comunicar com R4, mas ainda pode falar com qualquer outra máquina na mesma vlan (e fora 150.1.200.40)).

```
R1#p 150.1.234.5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.234.5, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/4 ms
R1#p 150.1.234.4

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.234.4, timeout is 2 seconds:
.....
Success rate is 0 percent (0/5)
R1#p 150.1.200.40

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.200.40, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/4 ms
```

Pode-se aplicar também uma MAC extended ACL para filtrarmos pacotes que **não são IP**:
http://www.cisco.com/en/US/docs/switches/lan/catalyst6500/ios/12.1E/native/command/reference/l1.html#wp1271486

VLAN MAPS - VACLS

A forma mais eficiente de se filtrar tráfego dentro de uma VLAN é utilizando uma Vlan ACL.

Digamos que temos os seguinte requerimentos de segurança para a rede da Figura 1:

- R1 tem permissão para se comunicar via IP dentro da subnet 150.1.234.0/24 apenas com R4 (234.4) ou R5 (234.5).
- R5 não pode se comunicar via IP com R3 dentro da subnet 150.1.234.0/24
- R3 não pode receber pacotes ARP do gateway.

*Notem que R3 não poderá comunicar com IPs fora da sua subnet uma vez que não terá em sua tabela ARP a entrada MAC para o IP do gateway.

Vamos aplicar esta Vlan Map no switch 1 (lembrando que essa vlan map apenas controla tráfego que entra ou sai da Vlan dentro deste mesmo switch):

```
vlan access-map vm_filter 10
action forward
match ip address 101
exit
```

```
!
vlan access-map vm_filter 20
action drop
match ip address 111
exit
!
vlan access-map vm_filter 30
action drop
match ip address 120
exit
vlan access-map vm_filter 40
action drop
match MAC address mc_arp
!
vlan access-map vm_filter 50
!
!
access-list 110 permit ip host 150.1.234.1 150.1.234.4 0.0.0.1
access-list 111 permit host 150.1.234.1 host 150.1.234.0 0.0.0.255
access-list 120 permit ip host 150.1.234.5 host 150.1.234.3
!
mac access-list extended mc_arp
permit host 0019.3065.c8c1 host 001b.2a53.6d58 0x806 0x0
```

Aplicando a Vlan Map a uma VLAN

Uma vez criada a Vlan Map aplica-se a VLAN desejada com o seguinte comando:

vlan filter *mapname* **vlan-list** *list*

vlan filter vm_filter vlan-list 300

Nota: Lembre-se que este tipo de filtro se aplica a uma Vlan e não a uma interface.

Em uma Vlan Map as ACL são utilizadas somente para definir qual tráfego será processado.

Isto é, permit = processar, deny = não processar.

O filtro é realizado pelo comando "**action**" onde podemos bloquear o tráfego com **drop** ou permitir com **forward**.

No exemplo acima, a ACL 110 corresponde a qualquer pacote IP com origem igual a 150.1.234.1 e com destino 150.1.234.4/31 (150.1.234.4 e 150.1.234.5) e a ação aplicada ao **vm_filter 10** foi **forward**, isto quer dizer que esse tráfego será permitido.

A ACL 111 corresponde a qualquer pacote IP com origem igual a 150.1.234.1 e qualquer destino dentro da subnet 150.1.234.0/24, a ação aplicada ao **vm_filter 20** foi **drop**, isto quer dizer que qualquer outro tráfego vindo de R1 nesta vlan será bloqueado.

```
R1#ping 150.1.234.4 rep 2

Type escape sequence to abort.
Sending 2, 100-byte ICMP Echos to 150.1.234.4, timeout is 2 seconds:
!!
Success rate is 100 percent (2/2), round-trip min/avg/max = 1/2/4 ms
R1#ping 150.1.234.5 rep 2

Type escape sequence to abort.
Sending 2, 100-byte ICMP Echos to 150.1.234.5, timeout is 2 seconds:
!!
Success rate is 100 percent (2/2), round-trip min/avg/max = 1/2/4 ms
R1#ping 150.1.234.3 rep 2

Type escape sequence to abort.
```

```
Sending 2, 100-byte ICMP Echos to 150.1.234.3, timeout is 2 seconds:
..
Success rate is 0 percent (0/2)
```

Nota: Este filtro não impede que R1 envia pacotes para o endereço de broadcast da vlan e receba respostas de R4 e R5. Uma vez que o filtro é especifico para os IPs 234.4 e 234.5

```
R1#ping 255.255.255.255 rep 1

Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 255.255.255.255, timeout is 2 seconds:

Reply to request 0 from 150.1.234.10, 1 ms
Reply to request 0 from 150.1.234.40, 4 ms
Reply to request 0 from 150.1.234.5, 1 ms
Reply to request 0 from 150.1.234.3, 1 ms
```

A ACL 120 corresponde a qualquer pacote IP com origem igual a 150.1.234.5 e destino 150.1.234.3; a ação aplicada à sequência **vm_filter 30** foi **drop**, isto quer dizer que esse tráfego será bloqueado.

```
R5#p 150.1.234.1 rep 2

Type escape sequence to abort.
Sending 2, 100-byte ICMP Echos to 150.1.234.1, timeout is 2 seconds:
!!
Success rate is 100 percent (2/2), round-trip min/avg/max = 1/2/4 ms
R5#p 150.1.234.3 rep 2

Type escape sequence to abort.
Sending 2, 100-byte ICMP Echos to 150.1.234.3, timeout is 2 seconds:
..
Success rate is 0 percent (0/2)
R5#p 150.1.234.4 rep 2

Type escape sequence to abort.
Sending 2, 100-byte ICMP Echos to 150.1.234.4, timeout is 2 seconds:
!!
Success rate is 100 percent (2/2), round-trip min/avg/max = 1/2/4 ms
R5#p 150.1.200.10 rep 2

Type escape sequence to abort.
Sending 2, 100-byte ICMP Echos to 150.1.200.10, timeout is 2 seconds:
!!
Success rate is 100 percent (2/2), round-trip min/avg/max = 1/1/1 ms
```

Na sequência **vm_filter 40** utilizamos uma MAC ACL para filtrar pacotes ARP (**0x806**) vindos do gateway (001b.2a53.6d58 = 150.1.234.10) para R3 (001b.2a53.6d58 = 150.1.234.3) e aplicamos a ação de **drop**.

```
R3#ping 150.1.234.4

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.234.4, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms
R3#ping 150.1.234.20 rep 2

Type escape sequence to abort.
Sending 2, 100-byte ICMP Echos to 150.1.234.20, timeout is 2 seconds:
!!
Success rate is 100 percent (2/2), round-trip min/avg/max = 1/1/1 ms
R3#ping 150.1.234.10 rep 2

Type escape sequence to abort.
```

```
Sending 2, 100-byte ICMP Echos to 150.1.234.10, timeout is 2 seconds:
..
Success rate is 0 percent (0/2)
R3#ping 150.1.200.10 rep 2

Type escape sequence to abort.
Sending 2, 100-byte ICMP Echos to 150.1.200.10, timeout is 2 seconds:
..
Success rate is 0 percent (0/2)
R3#sh arp | i 150.1.234.10
Internet 150.1.234.10 0 Incomplete ARPA
```

Podemos ver que R3 pode falar com outras máquinas na subnet 150.1.234.0/24 mas não pode se comunicar com seu gateway 234.10 por nao possue uma entrada ARP para este IP e por consequência também não pode falar fora de sua vlan.

vm_filter 50 foi criado para permitir qualquer outro trafego IP ou MAC dentro desta VLAN. Uma sequência "vazia" assume como padrão a ação de permitir qualquer tipo de tráfego (**action forward**).

Sem esta sequência a ação implícita em uma Vlan Map é a de bloquear todo tráfego.

Nota: Deve-se tomar cuidado especial ao utilizar MAC ACLs em um Vlan map, já que podemos acabar bloqueando BPDUs e impedir o correto funcionamento do Spanning-Tree (STP) e gerar loops de camada 2 na rede (broadcast storms).

Valores que podemos esperar ver num LAB de CCIE:

- **0x0806 = ARP**
- **lsap 0xAAAA = PVST+**
- **0x4242 = STP and PVST**
- **0x86DD = IPv6**

Outros valores de Ethernet Types podem ser encontrados no DOC CD da CISCO:
http://www.cisco.com/en/US/docs/ios/12_2/ibm/vol1/command/reference/br1fethc.html#wp1017386

Alguns detalhes sobre as VACLs:

- Esse tipo de ACL é configurada de forma semelhante aos **route-maps**. Trabalham de forma sequêncial. Isto é a ordem das entradas é importante.
- São processadas em hardware, não irão causar problemas de performance.
- Se utilizarmos uma vlan-map muito extensa o switch poderá levar mais tempo para realizar um boot.
- Não existe suporte para logging nestas ACLs.

Private-VLANs em CISCO CATOS

Afinal, a idéia das Private-Vlans surgiu primeiro no CATOS e foi depois implementada em IOS.

1o. Configurar o mode VTP como transparente:

COS
set vtp mode transparent
IOS
vtp mode transparent

2o. Criação das VLAN privada primária:

COS

set vlan *primary_number* **pvlan-type primary**
IOS
(global) **vlan** *primary_number*
(vlan-config) **private-vlan primary**

3o. Criação das VLANs secundárias:

COS
set vlan *secondary_number* **pvlan-type** [**isolated** | **community** | **twoway-community**]
IOS
(global) **vlan** *secondary_number*
(vlan-config) **private-vlan** [**isolated** | **community**]

4o. Associação da VLAN primária às secundárias:

COS
set pvlan *primary_number secondary_number*
IOS
(global) **vlan** *primary_number*
(vlan-config) **private-vlan association** *secondary_number_list* [**add** *secondary_number_list*]

5o. Configurar as portas conectadas a VLANs secundárias:

COS
set pvlan *primary_number secondary_number mod/port* [sc0]
IOS
(global) **interface** *type mod/port*
(interface) **switchport**
(interface) **switchport mode private-vlan host**
(interface) **switchport mode private-vlan host-association** *primary_number secondary_number*

6o. Configuração da porta promíscua:

COS
set pvlan mapping *primary_number secondary_number mod/port*
IOS
(global) **interface** *type mod/port*
(interface) **switchport**
(interface) **switchport mode private-vlan promiscuous**
(interface) **switchport mode private-vlan mapping** *primary_number secondary_number*

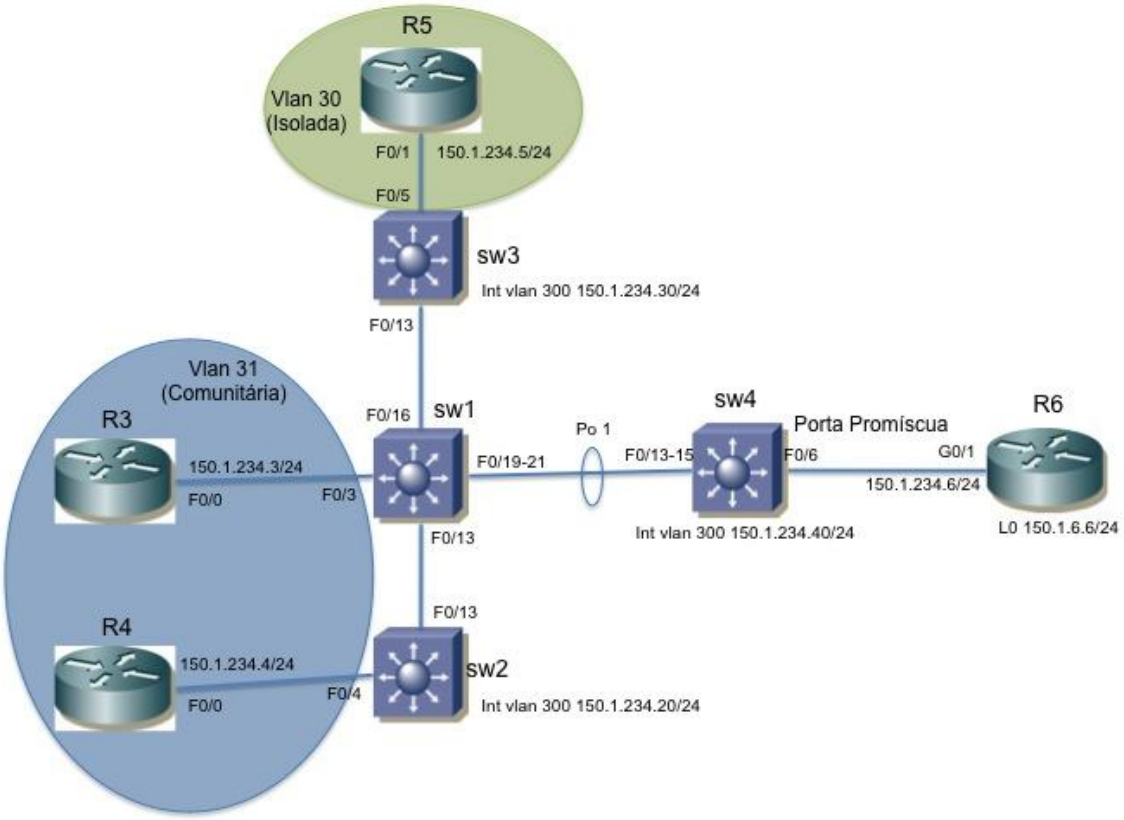
7o. (Opcional) Configurar interfaces L3 (SVI - switched virtual interfaces - IOS ou MSFC Multilayer Swich Feature Card - CATOS) nos switches para a realização do roteamento inter-VLAN:

COS
set pvlan mapping *primary_number secondary_number* **15/1**
session 15
(privileged)**config t**
(global)**interface vlan** *primary_number*
(interface)**ip address** *address mask*
IOS
(global) **interface** *primary_number*
(interface) **ip address** *address mask*
(interface) **private-vlan mapping** *primary_number secondary_number*

Verificação:

COS
show pvlan *number*
show pvlan mapping
show pvlan capability *mod/port*
IOS
show vlan private-vlan *[type]*
show interface private-vlan mapping
show interface *type mod/port* **switchport**

Implementando Private-Vlans em um Switch CISCO



Private-VLANs foram criadas com basicamente 2 propósitos:

- Prover segmentação entre pontos de rede sem a necessidade de se subdividir a rede em várias subnets. Em uma VLAN "tradicional" todos os pontos conectados a ela podem se comunicar entre si sem a necessidade de um elemento de L3 (que realiza routing). Caso exista a necessidade de se isolar esses pontos seria necessária a criação de outros segmentos e outras subnets o que exige mais endereços IPs (outras técnicas como "VLAN Access Maps" ou Protected ports também podem ser utilizadas, mas isso virá em outro post). Empresas de outsourcing de redes podem utilizar de Private-Vlans para isolar servidores de diferentes clientes em uma mesma VLAN sem a necessidade de criar um novo endereçamento ou sub-rede.
- Outro uso seria em Services Providers (ISPs), onde a limitação de 1005 VLANs ativas pode ser superada com a criação de "sub-VLANs", assim mais clientes podem ser conectados a uma rede metro-ethernet por exemplo.

Em resumo, uma Private-vlan segmenta uma VLAN em sub-domínios: uma vlan primária e múltiplas vlans secundárias.

Existem dois tipos de vlans secundárias:

- Vlans Comunitárias , onde as portas conectadas nesta vlan podem se comunicar entre si e com a porta promíscula, mas não se comunicam com nenhuma outra porta de outras vlans.
- Vlans Isoladas, onde as portas desta vlan apenas se comunicam com a porta promíscuaa.

Podem existir três tipos de portas em uma private-vlan:

- Porta Promíscua: Essa porta pertence a uma única VLAN primária e pode se comunicar com qualquer outra porta em qualquer VLAN. Normalmente configura-se portas ligadas a equipamentos L3 como portas promíscuas.
- Porta Isolada: Pertence a uma VLAN secundária e apenas se comunica com uma porta promíscua
- Porta Comunitária: Pertence a uma VLAN secundária e pode se comunicar com outras portas na mesma vlan secundária e com um porta promíscua.

Em resumo, a VLAN primária é usada para enviar os pacotes de um roteador (leia: qualquer dispositivo L3) para qualquer ponto da VLAN. Uma VLAN secundária isolada apenas envia pacotes entre um roteador e uma única máquina e uma VLAN secundária comunitária permite que um grupo de maquinas se comunique entre si e com um roteador (que poderá enviar os pacotes para qualquer outra porta dentro de qualquer outra VLAN).

Implementação:

*Foi utilizado nesse exemplo um rack similar ao usado no Internetwork Expert WB v4

Os passos 1 a 3 serão realizados em todos os switches (sw1, sw2, sw3, sw4), embora não seja necessário configurar as VLANs secundárias caso nenhuma porta esteja nesta VLAN)

1o: Configurar o mode VTP como transparente (é necessário para a criação de private-vlans)

```
vtp mode transparent
```

2o: Criação das VLANs

```
vlan 300
private-vlan primary
vlan 30
private-vlan isolated
vlan 31
private-vlan community
```

3o: Associação da VLAN primária às secundárias:

```
vlan 300
private-vlan association 30-31
```

4o: Configurar as portas conectadas a VLANs secundárias

```
sw1
interface FastEthernet0/3
switchport private-vlan host-association 300 31
switchport mode private-vlan host

sw2
interface FastEthernet0/4
```

```
switchport private-vlan host-association 300 31
switchport mode private-vlan host

sw3
interface FastEthernet0/5
switchport private-vlan host-association 300 30
switchport mode private-vlan host
```

Nota: Caso a VLAN primária não seja associada a VLAN secundária as portas configuradas como host nestas VLANs secundárias passaram para o estado de up down
ex:

```
sw2#sh int f0/4 desc
Interface Status Protocol Description
Fa0/4 up down

sw1#sh vlan private-vlan

Primary Secondary Type Ports
------- --------- ----------------- ------------------------------------------
300 30 isolated Fa0/1
300 31 community Fa0/3

sw2#sh vlan private-vlan

Primary Secondary Type Ports
------- --------- ----------------- ------------------------------------------
300 30 isolated
300 31 community Fa0/4

sw3#sh vlan private-vlan

Primary Secondary Type Ports
------- --------- ----------------- ------------------------------------------
300 30 isolated Fa0/5
300 31 community

sw4#sh vlan private-vlan

Primary Secondary Type Ports
------- --------- ----------------- ------------------------------------------
300 30 isolated Fa0/6
300 31 community Fa0/6
```

Nota: F0/6 por ser uma porta promíscua pertence a ambas VLANs

5o: Configuração da porta promíscua (neste exemplo ligada ao R6)

```
sw4
interface FastEthernet0/6
switchport private-vlan mapping 300 30-31
switchport mode private-vlan promiscuous
```

6o: (Opcional) Configurar interfaces L3 (SVI - switched virtual interfaces) nos switches para a realização do roteamento inter-VLAN.

```
sw3
interface Vlan300
ip address 150.1.234.30 255.255.255.0
private-vlan mapping 30-31

sw4
```

```
interface Vlan300
ip address 150.1.234.40 255.255.255.0
private-vlan mapping 30-31
```

As portas entre os switches foram configuradas como trunk.

```
interface FastEthernet0/13
switchport trunk encapsulation dot1q
switchport mode trunk
```

E apenas como extra, 3 portas entre o sw1 e sw4 foram configuradas como um port-channel L2 em trunk.

```
interface range FastEthernet0/19 - 21
switchport trunk encapsulation dot1q
switchport mode trunk
channel-group 1 mode active

interface Port-channel1
switchport trunk encapsulation dot1q
switchport mode trunk
```

Também com extra para testar conectividade, neste LAB implementei uma rota default no R5 apontando para R6 150.1.234.6 e uma interface loopback 0 em R6 com endereço 150.1.6.6/24

```
r6
int lo0
ip add 150.1.6.6 255.255.255.0

r5
ip route 0.0.0.0 0.0.0.0 150.1.234.6
```

Verificação:

R3 pertence a uma VLAN secundária comunitária, portanto só poderá se comunicar com R4 que também está na VLAN secundária comunitária, e com R6 e com as SVIs dos sw3 e sw4

Fazendo um ping para um endereço de broadcast desta subnet 150.1.234.0/24, temos o seguinte resultado.

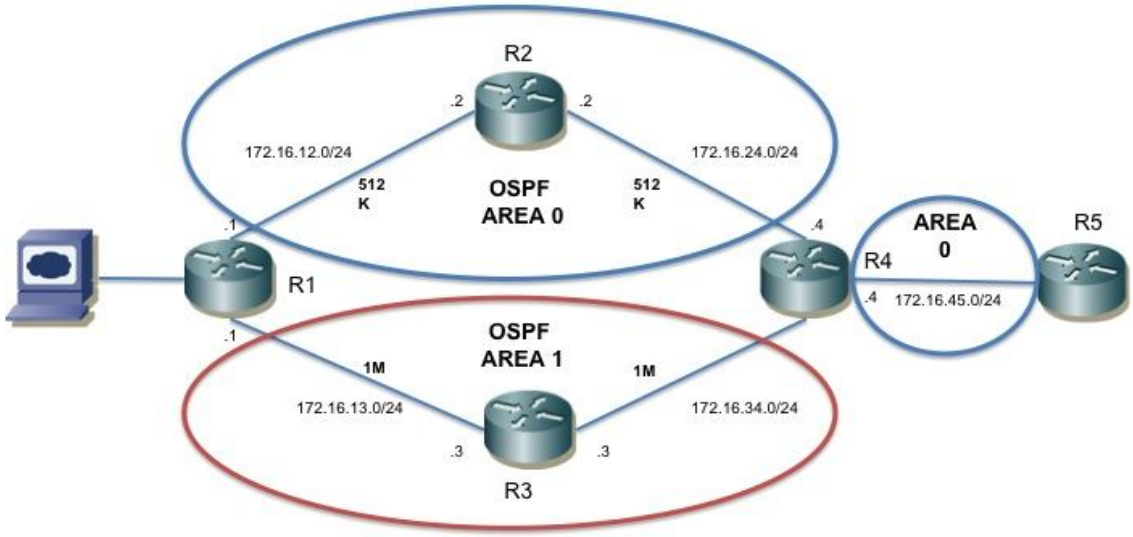
```
R3#ping 150.1.234.255

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.234.255, timeout is 2 seconds:

Reply to request 0 from 150.1.234.40, 1 ms
Reply to request 0 from 150.1.234.4, 1 ms
Reply to request 0 from 150.1.234.6, 1 ms
Reply to request 0 from 150.1.234.30, 1 ms
(...)

R4#ping 150.1.234.255

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.234.255, timeout is 2 seconds:

Reply to request 0 from 150.1.234.40, 1 ms
Reply to request 0 from 150.1.234.3, 1 ms
```

```
Reply to request 0 from 150.1.234.6, 1 ms
Reply to request 0 from 150.1.234.30, 1 ms

R5#ping 150.1.234.255

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.234.255, timeout is 2 seconds:

Reply to request 0 from 150.1.234.40, 1 ms
Reply to request 0 from 150.1.234.6, 1 ms
Reply to request 0 from 150.1.234.30, 1 ms

R5#ping 150.1.6.6

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.6.6, timeout is 2 seconds:

Reply to request 0 from 150.1.234.6, 1 ms
Reply to request 0 from 150.1.234.6, 1 ms
Reply to request 0 from 150.1.234.6, 1 ms
Reply to request 0 from 150.1.234.6, 1 ms
```

Design de Rede - roteando com OSPF – troubleshooting

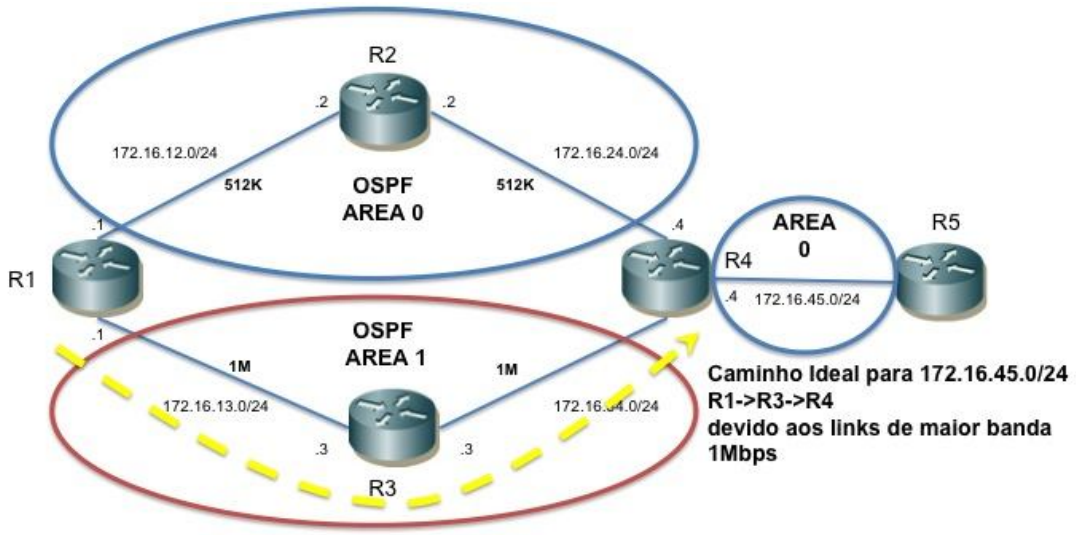
Um engenheiro de rede estava tendo dificuldades em rotear o tráfego pelos links de maior banda do site que ele mesmo havia "desenhado".

A parte da rede que importa neste problema era mais ou menos como a rede abaixo:



O problema era que os usuários conectados ao roteador R1 estavam roteando pelos links de menor banda (512k) para chegar a redes conectadas ao roteador R5.

O que ele desejava era rotear da seguinte forma:



Seguindo os links de maior banda 1Mbps

Fica claro pelo desenho da rede que isso não ia ser possível!

Vamos analisar a tabela de roteamento do roteador R1:

```
r1#sir | b Gate
Gateway of last resort is not set

172.16.0.0/24 is subnetted, 5 subnets
O 172.16.45.0 [110/391] via 172.16.12.2, 00:01:57, Ethernet0/0
O 172.16.34.0 [110/200] via 172.16.13.3, 00:01:57, Ethernet0/1
O 172.16.24.0 [110/390] via 172.16.12.2, 00:01:57, Ethernet0/0
C 172.16.12.0 is directly connected, Ethernet0/0
C 172.16.13.0 is directly connected, Ethernet0/1
```

Agora se analisarmos os custos dos links nos roteadores R2 e R3 vamos ver q os custos são menores no caminho R1 -> R3 -> R4.

```
r2#sh ip ospf interface br
Interface PID Area IP Address/Mask Cost State Nbrs F/C
Et0/1 1 0 172.16.24.2/24 195 BDR 1/1
Et0/0 1 0 172.16.12.2/24 195 DR 1/1

r3#sh ip ospf interface br
Interface PID Area IP Address/Mask Cost State Nbrs F/C
Et0/0 1 1 172.16.13.3/24 100 DR 1/1
Et0/1 1 1 172.16.34.3/24 100 BDR 1/1
```

Então, porque R1 estava escolhendo o pior caminho para chegar até a rota 172.16.45.0/24?
Vamos matar um dos links no R2 somente para verificar que a rota pelo caminho de maior banda existe:

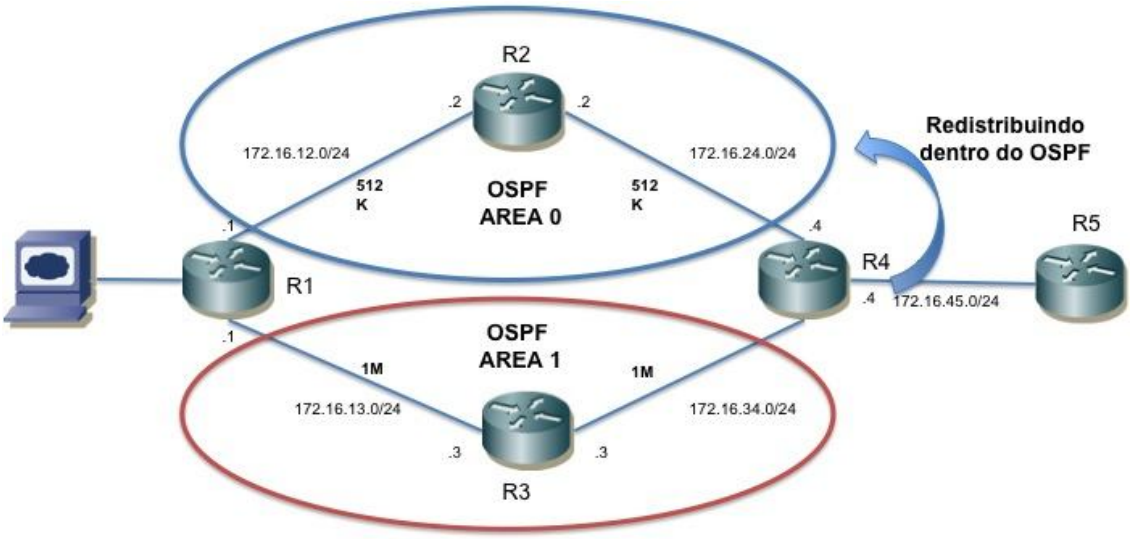
```
r1(config)#int e0/0
r1(config-if)#shut
r1(config-if)#
*Mar 1 00:17:28.387: %OSPF-5-ADJCHG: Process 1, Nbr 172.16.24.2 on Ethernet0/0 from FULL
to DOWN, Neighbor Down: Interface down or detached
r1(config-if)#end
r1#
*Mar 1 00:17:30.371: %LINK-5-CHANGED: Interface Ethernet0/0, changed state to
administratively down
*Mar 1 00:17:30.587: %SYS-5-CONFIG_I: Configured from console by console
*Mar 1 00:17:31.371: %LINEPROTO-5-UPDOWN: Line protocol on Interface Ethernet0/0,
changed state to down
r1#sh ip route | b Gatew
Gateway of last resort is not set
```

```
172.16.0.0/24 is subnetted, 5 subnets
O IA 172.16.45.0 [110/201] via 172.16.13.3, 00:00:08, Ethernet0/1
O 172.16.34.0 [110/200] via 172.16.13.3, 00:00:08, Ethernet0/1
O IA 172.16.24.0 [110/395] via 172.16.13.3, 00:00:08, Ethernet0/1
O IA 172.16.12.0 [110/590] via 172.16.13.3, 00:00:08, Ethernet0/1
C 172.16.13.0 is directly connected, Ethernet0/1
```

Podemos ver que o custo para essa rede pelo caminho do roteador R3 é de apenas 201 , enquanto pelo R2 o custo (maior) é de 391.

Então, por que R1 não está roteando da maneira desejada?

Vamos ver como a situação muda ao redistribuirmos a rota dentro do OSPF e não mais divulgarmos a mesma diretamente com o comando "**network**"




```
r4(config)#router ospf 1
r4(config-router)#no network 172.16.45.4 0.0.0.0 area 0
r4(config-router)#redistribute connected subnets metric-type 1
r4(config-router)#end

r1#sh ip route
(...)
172.16.0.0/24 is subnetted, 5 subnets
O E1 172.16.45.0 [110/220] via 172.16.13.3, 00:00:00, Ethernet0/1
O 172.16.34.0 [110/200] via 172.16.13.3, 00:00:00, Ethernet0/1
O 172.16.24.0 [110/390] via 172.16.12.2, 00:00:00, Ethernet0/0
C 172.16.12.0 is directly connected, Ethernet0/0
C 172.16.13.0 is directly connected, Ethernet0/1
r1#traceroute 172.16.45.4

Type escape sequence to abort.
Tracing the route to 172.16.45.4

1 172.16.13.3 96 msec 140 msec 96 msec
2 172.16.34.4 116 msec * 140 msec
```

Agora o tráfego segue pelo caminho de maior banda! Ficou mais fácil de entender o que está acontecendo!

Parte 3 : Solução

De acordo com a RFC 2328 seção 11, a ordem de preferência para as rotas OSPF é:

- rotas intra-area, O
- rotas interarea, O IA
- rotas externas tipo 1, O E1
- rotas externas tipo 2, O E2

Essa ordem não pode ser mudada dentro do mesmo processo OSPF e por isso uma rota do tipo O IA (area 1, no nosso exemplo) nunca vai ter preferência (mesmo com um menor custo) sobre uma rota do tipo intra-area (dentro da area 0 no nosso exemplo).

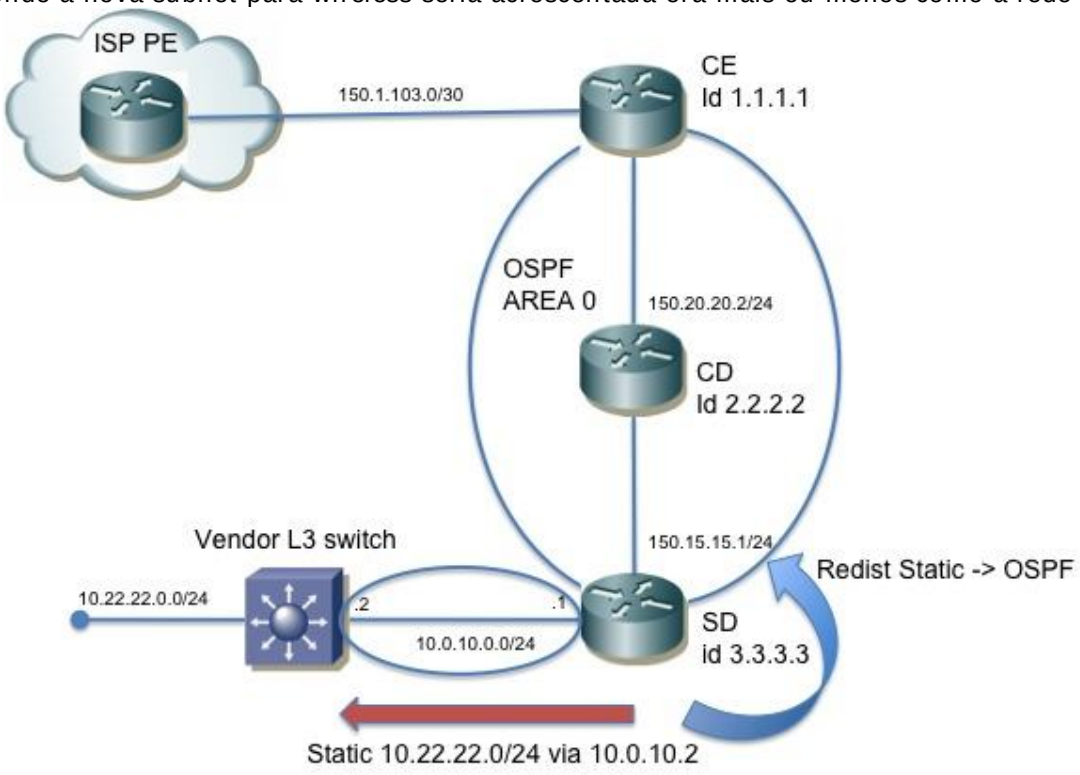
Uma forma alternativa seria criar mais um processo de OSPF no mesmo router e trabalhar com distância administrativa.

Mas idealmente, o melhor é evitar desenhar a rede de forma que este tipo de problema aconteça.

Troubleshooting avançado em Roteamento com OSPF em roteadores Cisco

Essa rede havia sido suportada por um time local durante muitos anos, e digamos, existiam algumas (muitas) configurações "estranhas" na rede.

A rede onde a nova subnet para wireless seria acrescentada era mais ou menos como a rede abaixo:



Os WAPs (wireless access points ou pontos de acesso sem-fio) estavam conectados a um switch L3. As redes existentes eram alcançáveis via rotas estáticas (mesmo existindo OSPF na rede).
configuradas nos roteadores SD, CD e CE.

A idéia inicial foi, porque não redistribuir as rotas estáticas no roteador SD dentro do OSPF, e evitar aquele tanto de rotas estáticas? É isto que foi realizado.

```
sd#sh ip route | b Gat
Gateway of last resort is not set

10.0.0.0/24 is subnetted, 2 subnets
C 10.0.10.0 is directly connected, Ethernet0/1
S 10.22.22.0 [1/0] via 10.0.10.2
150.15.0.0/24 is subnetted, 1 subnets
C 150.15.15.0 is directly connected, Ethernet0/0
150.20.0.0/24 is subnetted, 1 subnets
O 150.20.20.0 [110/20] via 150.15.15.2, 00:02:05, Ethernet0/0
sd#sh ip route 10.22.22.0
Routing entry for 10.22.22.0/24
Known via "static", distance 1, metric 0
Redistributing via ospf 1
Advertised by ospf 1 metric-type 1 subnets
Routing Descriptor Blocks:
* 10.0.10.2
Route metric is 0, traffic share count is 1
```

No roteador CD:

```
cd#sh ip route | i 10.22.22.0
O E1 10.22.22.0 [110/40] via 150.15.15.1, 00:09:49, Ethernet0/1
cd#sh ip route 10.22.22.0
Routing entry for 10.22.22.0/24
Known via "ospf 1", distance 110, metric 40, type extern 1
Last update from 150.15.15.1 on Ethernet0/1, 00:09:57 ago
Routing Descriptor Blocks:
* 150.15.15.1, from 3.3.3.3, 00:09:57 ago, via Ethernet0/1
Route metric is 40, traffic share count is 1
```

No roteador CE:

```
ce#sh ip route | i 10.22.22.0
ce#sh ip route 10.22.22.0
% Subnet not in table
```

E por algum motivo a rota 10.22.22.0/24 não estava na tabela de roteamento do roteador CE.

Basta verificar a dababase do OSPF para confirmar se a rede estaria sendo divulgada normalmente dentro do OSPF. Como era uma rota externa (redistribute static), usei o comando:
show ip ospf database external, e pude verificar que a rede estava presente.

```
ce#sh ip ospf database external

OSPF Router with ID (1.1.1.1) (Process ID 1)

Type-5 AS External Link States

LS age: 1256
Options: (No TOS-capability, DC)
LS Type: AS External Link
Link State ID: 10.22.22.0 (External Network Number )
Advertising Router: 3.3.3.3
LS Seq Number: 80000003
Checksum: 0x9432
Length: 36
Network Mask: /24
Metric Type: 1 (Comparable directly to link state metric)
TOS: 0
Metric: 20
Forward Address: 10.0.10.2
External Route Tag: 0
```

Então, porque a rota não estava sendo instalada na tabela de roteamento?

Verificando a tabela completa de roteamento do roteador CE temos:

```
ce#sh ip route | b Gate
Gateway of last resort is not set

10.0.0.0/24 is subnetted, 1 subnets
S 10.0.10.0 [1/0] via 150.20.20.2
150.15.0.0/24 is subnetted, 1 subnets
O 150.15.15.0 [110/20] via 150.20.20.2, 00:58:36, Ethernet0/0
150.20.0.0/24 is subnetted, 1 subnets
C 150.20.20.0 is directly connected, Ethernet0/0
```

E aí podemos ver a causa do problema, pois em resumo, o problema aqui são rotas externas ao OSPF - redistribuídas - não sendo colocadas na tabela de roteamento.

Trata de um problem com o FA - forwarding address - "endereço de encaminhamento!?"

Esse valor é acrescentado aos LSAs externos (LSA tipo 5) para indicar para onde o tráfego deve ser enviado. Isso torna em alguns casos o roteamento mais eficiente dentro do OSPF.

Link externo: Os efeitos do FA nos LSAs tipo 5 na seleção de caminhos (rotas) no OSPF

O FA pode ser ou 0.0.0.0 ou qualquer outro IP.

Caso, seja 0.0.0.0 o pacotes são enviados ao router que originou o LSA ou seja o ASBR (Autonomous System Boundary Router).

Existem condicões certas para que um FA seja o next-hop (ou proxima salto) da rota redistribuida, ao invés de 0.0.0.0:

- OSPF habilitado no ASBR na interface onde o next hop está configurado.
- Essa interface não está como passiva no OSPF
- Essa interface não ser do tipo ponto-a-ponto ou ponto-multiponto (no OSPF)

Uma das regras que temos no OSPF (rfc2328) é que este endereço deve ser roteado via OSPF interno (inter (O IA) ou intra-area (tipo O) para que a rede divulgada no LSA tipo 5 possa ser usada, isto é , para que ela apareça na tabela de roteamento. E esse é uma das regras pouco conhecidas do OSPF, por incrível que pareça.

Da RFC 2328:
"If the forwarding address is non-zero, look up the forwarding address in the routing table.[24] The matching routing table entry must specify an **intra-area or inter-area path**; if no such path exists, do nothing with the LSA and consider the next in the list."

No nosso exemplo vemos que o FA para a rede 10.22.22.0/24 é 10.0.10.2.

E se revermos a tabela do router CE, vemos que a rota para 10.0.10.0/24 é uma rota estática:

```
ce#sh ip route | b Gate
Gateway of last resort is not set

10.0.0.0/24 is subnetted, 1 subnets
S 10.0.10.0 [1/0] via 150.20.20.2
```

Essa rota, fazia parte do monte de rotas estáticas configurados em meio a BGP, OSPF, RIP nessa rede. O que, por fim, causou o problema.

Uma vez retirada essa rota estática da tabela e substituindo a mesma por uma rota tipo O (intra-area OSPF), a rota 10.22.22.0/24 apareceu na tabela como uma rota O E1, como esperado.

```
ce(config)#no ip route 10.0.10.0 255.255.255.0 150.20.20.2
ce(config)#end
ce#sh ip route | b Gateway
Gateway of last resort is not set

10.0.0.0/24 is subnetted, 2 subnets
O 10.0.10.0 [110/30] via 150.20.20.2, 00:00:00, Ethernet0/0
O E1 10.22.22.0 [110/50] via 150.20.20.2, 00:00:00, Ethernet0/0
150.15.0.0/24 is subnetted, 1 subnets
O 150.15.15.0 [110/20] via 150.20.20.2, 00:00:00, Ethernet0/0
150.20.0.0/24 is subnetted, 1 subnets
C 150.20.20.0 is directly connected, Ethernet0/0
```

Uma questão de verificar a tabela do OSPF depois a de roteamento e resolver o problema.

A CISCO disponibiliza nas suas páginas vários fluxogramas que ajudam neste tipo de solução de problemas. Neste caso, uma pesquisa básica no google: troubleshooting ospf cisco

Link: Troubleshooting OSPF

Como o problema é de roteamento vamos ao link interno:

Link: Troubleshooting the ospf routing table

E como o problema é com uma rota externa, seguimos para o link:

Link: Troubleshooting External Link-State Advertisements

Então, é apenas uma questão de seguir o fluxo:

- O LSA externo existe na tabela do OSPF? **SIM** (show ip ospf database external)

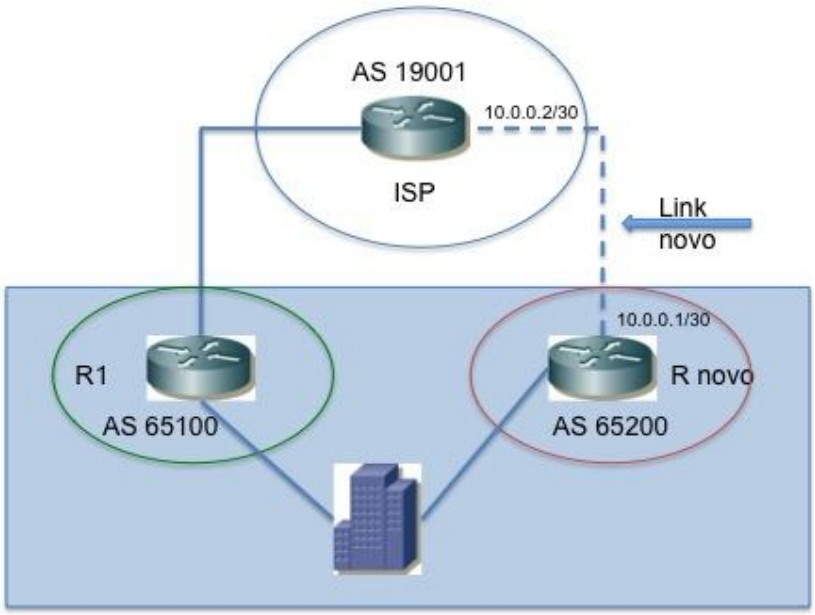
- O FA é 0.0.0.0 -> **NÃO**

- O FA é conhecido via OSPF inter ou intra-area -> **NÃO**

E chegamos a resposta: **FA deve ser alcançado via uma rota inter ou intra-area.**

BGP - Salvando uma "change"!

A change (mudança) consistia na instalação de um roteador novo em um site e a conexão do mesmo com a LAN e WAN (ISP). (Na figura abaixo - R-novo)



A change foi marcada e tudo acertado.

A parte da configuração que importa nesse post era a seguinte:

R novo:
```
router bgp 65200
no synchronization
bgp log-neighbor-changes
neighbor 10.0.0.2 remote-as 19001
```

ISP:
```
router bgp 19001
no synchronization
bgp log-neighbor-changes
neighbor 10.0.0.1 remote-as 65200
```

Deu-se início a change e, para a minha supresa, não foi possível fechar a seção de BGP com o roteador do Provedor. A seguinte mensagem começou a aparecer no roteador novo:

%BGP-3-NOTIFICATION: received from neighbor 10.0.0.2 2/2 (peer in wrong AS) 2 bytes FEB0

Claramente, havia algo errado entre os números de AS usado na configuração. (FEB0 = 65200 em decimal). Como o meu lado estava "certo". Contactei o Provedor para confirmar a configuração do lado dele. E estava lá o problema. A configuração era a seguinte:

ISP
```
router bgp 19001
no synchronization
bgp log-neighbor-changes
neighbor 10.0.0.1 remote-as 65100
```

Devido a um problema de comunicação entre os times, eles haviam configurado o AS antigo, ao invés de usar o AS novo 65200.

Quem já trabalhou com grandes provedores e em empresas com um sistema de changes bem restrito sabe que, às vezes, algo simples como dizer para eles: "muda a configuração do seu roteador para usar o AS 65200" pode ser impossível. Primeiro que neste caso, não seria somente uma linha. Existem outras configurações que eu não postei que também envolviam esse número (como filtros com as-path aplicados a route-maps, etc.). Em segundo é que nunca se sabe quem estará acompanhando a change do outro lado.

Por fim, não era possível realizar a mudança de configuração e a migração do site para o router/link novo teria que ser cancelada.

neighbor {*ip-address*| *peer-group-name*} local-as *as-number*

http://www.cisco.com/en/US/docs/ios/12_2/iproute/command/reference/1rfbgp1.html#wp1185470

Esse comando é usado para se alterar o número AS local enviado nos pacotes de BGP para o vizinho. É uma forma de "enganar" o vizinho fingindo-se ter um AS diferente do realmente configurado.

Então apliquei a configuração:

```
r-novo(config)#router bgp 65200
r-novo(config-router)#neighbor 10.0.0.2 local-as 65100

%BGP-5-ADJCHANGE: neighbor 10.0.0.2 Up
```

Change Salva!!!!

O BGP sabe apesar de ter feito a conexão e a migração ter acontecido, existe um problema com comando. Verificando a tabela de BGP do roteador do Provedor, podemos ver o problema:

```
isp-r#sh ip bgp | b Netw
Network Next Hop Metric LocPrf Weight Path
*> 10.0.55.0/24 10.0.0.1 0 0 65100 65200 i
```

O AS falso foi acrescentado ao AS PATH das redes vinda do roteador novo!

E isso pode gerar problemas de roteamento indevido, uma vez que a quantidade de ASs em um AS PATH faz parte da decisão do BGP na hora de escolher a melhor rota para um site.

http://www.cisco.com/en/US/tech/tk36...80094431.shtml

Cada site dessa empresa tinha 2 links. O link com maior banda sempre ia por um caminho com um menor numero de AS. Assim outros sites utilizavam esse caminho para alcançar o demais sites.

Como neste caso um novo AS foi acrescentado, gerou-se um problema onde o caminho para esse site agora passou a ser o pior link (com menos banda).

Manipulando a decisão do BGP em outra parte da rede, foi possível corrigir esse problema, enquanto aguardávamos uma nova change para deixar tudo no ponto (isto é mudar o AS na configuração do BGP para 65200) e eliminar esse "gato" na rede!

Kron - agendando comandos no IOS CISCO - (filtro BGP time-based)

O comando usado é o "KRON".
http://www.cisco.com/en/US/docs/ios/12_3/configfun/command/reference/cfr_1g04.html#wp1049808

Esse comando é uma versão bem simplificada do "cron" que existe no EEM da CISCO e que é muito mais versátil e complexo.

O Kron, basicamente, permite a execução de comandos EXEC uma única vez, ou, em intervalos específicos, em determinado horario, ou (dependendo da versao de IOS) após o ínicio do sistema (system-startup depois de um boot).

Não é possível usar o KRON com comandos que exigem qualquer tipo de interação com o usuário (como reload, clear counters, copy running-config startup-config, etc).

A configuração do KRON consiste em duas etapas:

- A definição de uma lista de comandos: **kron policy-list** *list-name*
- A definição de quando o a lista de comandos será executada: **kron occurrence** *occurrence-name*

O uso deste comando é restrito, mas alguns exemplos são:

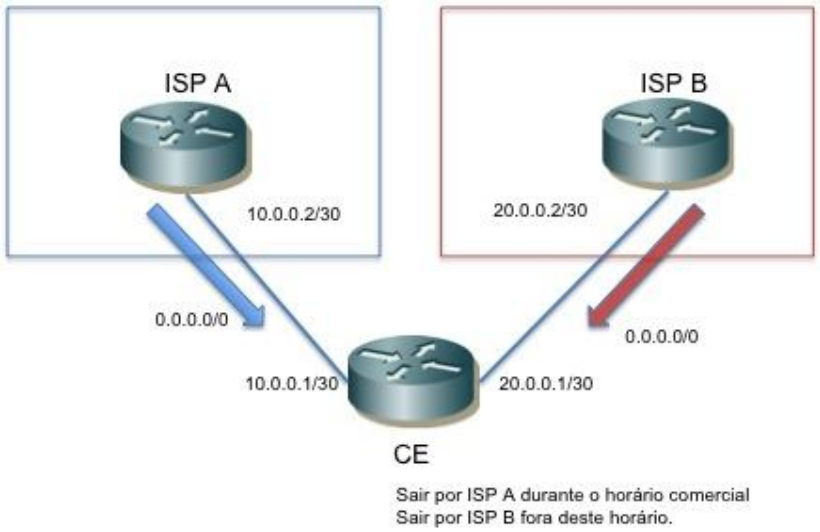
Realizar o backup da configuração dos roteadores todo domingo as 23:00

```
r1(config)# kron policy-list Backup
r1(config-kron-policy)# show run | redirect tftp://192.168.1.1/r1.cfg
r1(config-kron-policy)# exit

r1(config)# kron occurrence Backup at 23:00 Sun recurring
r1(config-kron-occurrence)# policy-list Backup
```

Salvar a configuração dos roteadores toda segunda as 04:00

```
kron policy-list SaveConfig
cli write
exit
kron occurrence SaveConfig at 04:00 Mon recurring
policy-list SaveConfig
r1#sh kron schedule
Kron Occurrence Schedule
kr_save inactive, will run again in 1 days 09:11:41 at 4 :00 on Mon
```

Imagine uma rede conectada a 2 ISPs.



Ambos estão enviando uma rota padrão (0.0.0.0/0) via BGP para o roteador CE do site. A tarefa a realizar seria a de somente utilizar a rota padrão vinda do ISP A durante o horário comercial e a rota vinda do ISP B fora do horário comercial (horário comercial : seg a sexta de 06:00 as 18:00).

A primeira coisa que pensamos é: fácil, temos apenas que utilizar uma Time-Based ACL (ACL baseada em tempo) e aplicar um filtro no BGP utilizando esta ACL. **Perfeito!**

Antes de utilizarmos qualquer filtro temos a seguinte tabela de BGP no roteador CE, onde vemos a rota default vinda de ambos provedores 10.0.0.2 e 20.0.0.2.
Nota: Usei um peso (weight) para que a rota do ISP A seja sempre preferida , caso ela exista na tabela BGP.

```
ce#sib
BGP table version is 2, local router ID is 20.0.0.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

Network Next Hop Metric LocPrf Weight Path
* 0.0.0.0 20.0.0.2 0 0 3 i
*> 10.0.0.2 0 500 2 i
```

Configuramos uma ACL baseada em tempo que estará ativa durante os finais de semana e fora do horário entre 6:00h e 18:00h.

```
ip access-list extended acl_bgp_f
deny ip any any time-range n_comercial
permit ip any any
!
time-range n_comercial
periodic weekend 0:00 to 23:59
periodic weekdays 0:00 to 5:59
periodic weekdays 18:00 to 23:59
```

Usamos essa ACL em um filtro de BGP (distribute-list in) para o vizinho no ISP A. Desta forma, bloquearemos a rota 0.0.0.0/0 vinda do ISP A fora do horário comercial e rotearemos por ISP B. No roteador CE configuramos:

```
router bgp 1
neighbor 10.0.0.2 distribute-list acl_bgp_f in
```

Mas, existe um problema!!!

O BGP não irá re-processar os filtros aplicados aos seus vizinhos, sem que exista alguma mudança nas rotas. Isto é, mesmo que o filtro mude, não haverá um novo update das rotas na tabela de BGP.

Então, como resolver esse problema? Precismos de fazer com que o BGP reprocesse as rotas.

O comando **clear ip bgp * soft** pode ser usado para reprocessar os filtros do BGP sem interromper as seções criadas (daí o nome soft).
http://www.cisco.com/en/US/docs/ios/...html#wp1249715

Vamos ver como fica, irei alterar o horário com o comando **clock** para ativar a ACL:

```
ce#sh ip bgp
BGP table version is 2, local router ID is 20.0.0.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

Network Next Hop Metric LocPrf Weight Path
* 0.0.0.0 20.0.0.2 0 0 3 i
*> 10.0.0.2 0 500 2 i
ce#sh clock
13:09:26.043 UTC Mon Feb 16 2009
ce#clock set 21:00:00 16 feb 2009
ce#clear ip bgp * soft
ce#sh ip bgp
BGP table version is 3, local router ID is 20.0.0.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

Network Next Hop Metric LocPrf Weight Path
*> 0.0.0.0 20.0.0.2 0 0 3 i
```

Agora a única coisa a fazer é agendar uma tarefa para os horários desejados e aplicar o comando "**clear ip bgp * soft**" assim iremos forçar o processo BGP a re-processar os filtros aplicados no horário que desejamos.

Criado dois processos, um que será executado todo dia as 6:00h e outro que será executado todos os dias as 18:00h

```
kron occurrence krt at 6:00 recurring
policy-list kr_bgp
!
kron occurrence krt1 at 18:00 recurring
policy-list kr_bgp1
!
kron policy-list kr_bgp
cli clear ip bgp * soft
!
kron policy-list kr_bgp1
cli clear ip bgp * soft
```

Verificando com "debug kron"

```
ce#sh clock
06:06:35.139 UTC Mon Feb 16 2009
ce#sh ip bgp
BGP table version is 3, local router ID is 20.0.0.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

Network Next Hop Metric LocPrf Weight Path
* 0.0.0.0 10.0.0.2 0 0 2 i
*> 20.0.0.2 0 0 3 i
ce#clock set 17:59:30 16 feb 2009
ce#
Feb 16 17:59:30.000: Clock Set Seen
Feb 16 17:59:30.000: Major 4, Minor 9
Feb 16 17:59:30.003: Start timer for krt
Feb 16 17:59:30.003: Start timer for krt2
ce#
Feb 16 18:00:30.003: Major 1, Minor 0
Feb 16 18:00:30.003: Timer Event krt2
Feb 16 18:00:30.003: Kron delay for next krt2 86400000
Feb 16 18:00:30.007: Call parse_cmd 'clear ip bgp * soft'
Feb 16 18:00:30.007: Kron CLI return 0
''
Feb 16 18:00:30.007: Major 4, Minor 7
Feb 16 18:00:30.007: Respond to end of CLI Process
ce#sib
BGP table version is 3, local router ID is 20.0.0.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

Network Next Hop Metric LocPrf Weight Path
*> 0.0.0.0 20.0.0.2 0 0 3 i
ce#sh access-list
Extended IP access list acl_bgp_f
10 deny ip any any time-range n_comercial (active) (1 match)
20 permit ip any any (1 match)

ce#clock set 05:59:30 16 feb 2009
ce#
```

```
Feb 16 05:59:30.000: Clock Set Seen
Feb 16 05:59:30.000: Major 4, Minor 9
Feb 16 05:59:30.003: Start timer for krt
Feb 16 05:59:30.003: Start timer for krt2
ce#
Feb 16 06:00:30.003: Major 1, Minor 0
Feb 16 06:00:30.003: Timer Event krt
Feb 16 06:00:30.003: Kron delay for next krt 86400000
Feb 16 06:00:30.007: Call parse_cmd 'clear ip bgp * soft'
Feb 16 06:00:30.015: Kron CLI return 0
''
Feb 16 06:00:30.015: Major 4, Minor 7
Feb 16 06:00:30.019: Respond to end of CLI Process
ce#sib
BGP table version is 6, local router ID is 20.0.0.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

Network Next Hop Metric LocPrf Weight Path
*> 0.0.0.0 10.0.0.2 0 500 2 i
* 20.0.0.2 0 0 3 i
```

TCL shell - CISCO IOS - programando um roteador usando scripts!

O TCL (*Tool Command Language)* shell. Essa shell foi criada para permitir a execução de scripts TCL diretamente no IOS CISCO. Estes scripts, claro, usam e interagem diretamente com os comandos disponíveis no IOS. Existe também a possibilidade de criar o script e depois pré-compilar os mesmos, salvando-os na memória FLASH ou disco. Além disto, podem ser compartilhados por múltiplos usuários no mesmo roteador e ao mesmo tempo.

Um exemplo do uso dessa shell aparece na prova prática da CCIE RS onde, usualmente, pede-se conectividade total entre os equipamentos , isto é, cada IP da rede deve ser capaz de pingar qualquer outro IP.

Agora imaginem, testar conectividade de 10 equipamentos repletos de interfaces e IPs?
Pingar cada IP de cada equipamento, de dentro de cada equipamento?
E ainda verificar o que falhou e ir atrás do problema?

Tarefa complicada que o seguinte TCL script simplifica e muito. Basicamente, ele serve para filtrar a resposta do comando PING, e apenas imprimir na tela os IPs que não responderem com um ICMP echo-reply.

foreach -> cria uma loop de iteraçao com os IPs listados
[exec ping $ips timeout 3] -> pinga cada IP da lista e usa um timeout de 3 seg
string first "!!!" -> verifica se na string de saída do comando "ping IP" encontramos "!!!"
$result == -1 } { puts ".. $ips "}} verifica se o resultado for negativo (sem resposta do ping) imprimi na tela dois pontos (..) e o IP que não respondeu.

```
r1#tclsh
r1(tcl)#proc pingall {} {
+>(tcl)#foreach ips {
+>(tcl)#155.1.1.1
+>(tcl)#155.4.4.4
+>(tcl)#155.6.6.6}
+>(tcl)#{ set result [ string first "!!!" [exec ping $ips timeout 3 ] ]
+>(tcl)#if { $result == -1 } { puts ".. $ips " } }
+>(tcl)#}
r1(tcl)#pingall
.. 155.4.4.4
.. 155.6.6.6
```

Somente para relembrar a saída tipica de um ping e entender melhor o exemplo:

```
r1#ping 155.1.1.1 timeout 3
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.1.1, timeout is 3 seconds:
!!!!!
Success rate is 100 percent (5/5)
```

Assim como o script verificou os "!!!" na saída do comando, nada foi impresso na tela

E para um ping que não recebeu resposta:

```
r1#ping 155.4.4.4 timeout 3
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.4.4.4, timeout is 3 seconds:
.U.U.
Success rate is 0 percent (0/5)
```

O script não encontra "!!!" e imprime na tela ".. e o IP"

Usei este exato script durante meu estudo e no dia do LAB CCIE R&S. Após configurar todos os equipamentos, acessei cada um deles e com um **show ip interface br** (tomar cuidado para não perder ips secundários que não são listados na saída deste comando) criei uma lista de IPs e colei os mesmos no script. Depois, foi apenas uma questão de copiar colar e executar o script em cada equipamento, e conferir o resultado.

Nota: Quem escreveu este script foi um grande amigo (monstro dos scripts) e ex-colega de trabalho o Daniel Longhi.

Esse post foi apenas o primeiro sobre o TCL, uma vez que, essa ferramenta vem cada vez mais sendo utilizada na CISCO, por exemplo nos sistemas de VOZ (Interactive Voice Response (IVR) e Monitoramento (Embedded Event Manager (EEM)) e outros.

Existem incontáveis exemplos do uso dessa shell, para manipular saída de comandos, monitorar o roteador, enviar emails com saídas de comandos (show tech...) etc.

TCL SCRIPT:

```
tclsh
proc pingall {} {
foreach ips {
172.16.16.1
172.16.123.1
172.16.14.1
172.16.50.25} { set result [ string first "!!!" [ exec ping $ips ] ]
if { $result == -1 } { puts ".. $ips " } }
}
```

Transmitindo pacotes Broadcast de uma rede para outra - ip helper-address

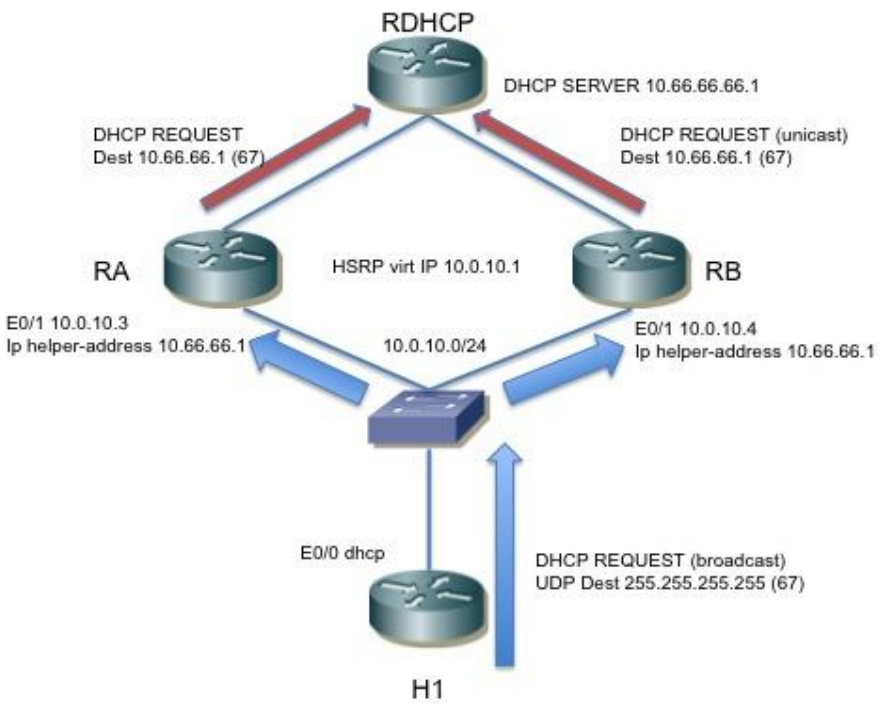
A pergunta tem por trás uma questão básica de redes:

Pacotes broadcast (destino 255.255.255.255) não são roteáveis por padrão. Isto é, se um roteador recebe um pacote com este destino , ele processa o pacote (ARP, DHCP, etc), mas não transmite este pacote para nenhum outro destino. Como o serviço de DHCP é feito utilizando pacotes UDP com destino broadcast, por padrão, seria necessário 1 servidor por subrede.

A solução: o comando: **ip helper-address** *ip*

http://www.cisco.com/en/US/docs/ios/12_3/ipaddr/command/reference/ip1_i1g.html#wp1084408

Esse comando permite ao roteador transmitir pacotes de broadcast para um (ou mais) destino escolhido.



Ex:
RA e RB

```
int e0/1
ip helper-address 10.66.66.1
```

Pacotes recebidos na interface E0/1 com destino 255.255.255.255 serão enviados para o destino 10.66.66.1 (como unicast).

Por padrão os seguintes protocolos são enviados:

- TFTP (port 69)
- DNS (port 53)
- Time (port 37)
- TACACS (port 49)
- BOOTP client (port 68)
- BOOTP server (port 67)
- NetBIOS name service (port 137)
- NetBIOS datagram service (port 138)

O controle de quais protocolos serão utilizados por este comando é feito por um outro comando:

ip forward-protocol{**udp**[*port*] | **nd**|**sdns**}

http://www.cisco.com/en/US/docs/ios/12_3/ipaddr/command/reference/ip1_i1g.html#wp1108053

Alguns exemplos:

```
LAB(config)#ip forward-protocol udp ?
<0-65535> Port number
```

```
biff Biff (mail notification, comsat, 512)
bootpc Bootstrap Protocol (BOOTP) client (68)
bootps Bootstrap Protocol (BOOTP) server (67)
discard Discard (9)
dnsix DNSIX security protocol auditing (195)
domain Domain Name Service (DNS, 53)
echo Echo (7)
isakmp Internet Security Association and Key Management Protocol (500)
mobile-ip Mobile IP registration (434)
nameserver IEN116 name service (obsolete, 42)
netbios-dgm NetBios datagram service (138)
netbios-ns NetBios name service (137)
netbios-ss NetBios session service (139)
non500-isakmp Internet Security Association and Key Management Protocol (4500)
ntp Network Time Protocol (123)
pim-auto-rp PIM Auto-RP (496)
rip Routing Information Protocol (router, in.routed, 520)
snmp Simple Network Management Protocol (161)
snmptrap SNMP Traps (162)
sunrpc Sun Remote Procedure Call (111)
syslog System Logger (514)
tacacs TAC Access Control System (49)
talk Talk (517)
tftp Trivial File Transfer Protocol (69)
time Time (37)
who Who service (rwho, 513)
xdmcp X Display Manager Control Protocol (177)
<cr>
```

Isto quer dizer que, por padrão, se configurarmos o comando ip-helper address em um interface, qualquer pacote de broadcast destes tipos serão enviados para o destino configurado.

No nosso exemplo, não somente os pacotes de DHCP, mas outros (às vezes, indesejáveis) serão também enviados ao servidor de DHCP. E enviar NETBIOs para o servidor DHCP, por exemplo, seria sem sentido.

Então, toda vez que configurarmos esse comando com o objetivo de enviarmos pacotes de DHCP para um servidor, é bom usarmos em conjunto o comando (global):

```
router(config)#ip forward-protocol udp 67
```

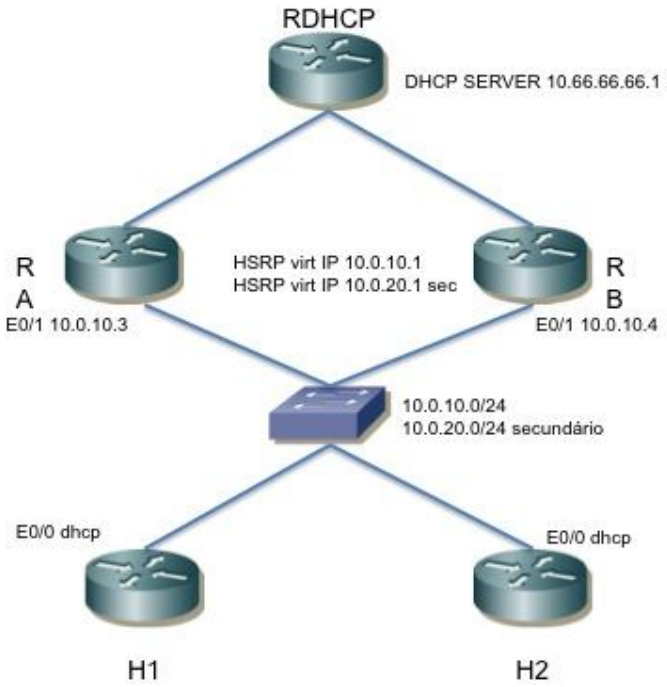
Como distribuir IPs via DHCP para endereços secundários em uma vlan? CISCO smart-r

Por padrão, quando um roteador CISCO recebe um pedido broadcast de DHCP (DHCPDISCOVER - udp 67) e está configurado com o comando **service dhcp** e a interface que recebeu o pacote está configurada com: **ip helper-address**, ele envia uma requisição unicast para o servidor de DHCP configurado, **mas apenas para a subnet do IP principal dessa interface.**

Isto é, mesmo que não exista mais IPs disponíveis, o roteador continuará procurando um IP nessa mesma subnet.

Vamos ver uma simulação!

Nota: Use HSRP nesse exemplo, mas neste caso não existe relevância.



Na figura acima, a interface e0/1 dos roteadores A e B foram configuradas da seguinte forma:

```
ra#sh run int e0/1
interface Ethernet0/1
ip address 10.0.20.3 255.255.255.0 secondary
ip address 10.0.10.3 255.255.255.0
full-duplex
standby 1 ip 10.0.10.1
standby 1 ip 10.0.20.1 secondary
standby 1 preempt
end

rb#sh run int e0/1
interface Ethernet0/1
ip address 10.0.20.4 255.255.255.0 secondary
ip address 10.0.10.4 255.255.255.0
ip helper-address 10.66.66.1
full-duplex
standby 1 ip 10.0.10.1
standby 1 ip 10.0.20.1 secondary
standby 1 preempt
end
```

Os roteadores H1 e H2 representam maquinas na rede que irão buscar endereços via DHCP

```
interface Ethernet0/0
ip address dhcp
full-duplex
```

10.66.66.1 é o endereço para onde os pacotes de broadcast recebidos serão enviados (como unicast).

Configurei um roteador como servidor de DHCP com a seguinte configuração:

```
ip dhcp excluded-address 10.0.10.1 10.0.10.253
ip dhcp excluded-address 10.0.20.1 10.0.20.10
!
```

```
ip dhcp pool main-pool
network 10.0.10.0 255.255.255.0
default-router 10.0.10.1
dns-server 4.2.2.1
domain-name vladrac.com
!
ip dhcp pool secund-pool
network 10.0.20.0 255.255.255.0
default-router 10.0.20.1
dns-server 4.2.2.1
domain-name vladrac.com
```

O pool para a rede primária 10.0.10.0/24 foi configurado com apenas 1 IP disponível : **10.0.10.254**.

E um segundo pool foi criado para a rede secundária na mesma Vlan: 10.0.20.0/24.

Podemos ver o problema analizando a saída de uns "debugs" no roteador rodando DHCP:

```
DHCPD: Sending notification of ASSIGNMENT:
DHCPD: address 10.0.10.254 mask 255.255.255.0
DHCPD: htype 1 chaddr cc03.0884.0000
DHCPD: lease time remaining (secs) = 86400
DHCPD: Appending default domain from pool
DHCPD: Using hostname 'h1.vladrac.com.' for dynamic update (from hostname option)
DHCPD: Sending DHCPACK to client 0063.6973.636f.2d63.6330.332e.3038.
3834.2e30.3030.302d.4574.302f.30 (10.0.10.254).
DHCPD: unicasting BOOTREPLY for client cc03.0884.0000 to relay 10.0.10.4.
dhcp#
DHCPD: Sending notification of DISCOVER:
DHCPD: htype 1 chaddr cc04.0884.0000
DHCPD: remote id 020a00000a001e0200000000
DHCPD: circuit id 00000000
DHCPD: DHCPDISCOVER received from client 0063.6973.636f.2d63.6330.342e.3038.
3834.2e30.3030.302d.4574.302f.30 through relay 10.0.10.4.
DHCPD: Seeing if there is an internally specified pool class:
DHCPD: htype 1 chaddr cc04.0884.0000
DHCPD: remote id 020a00000a001e0200000000
DHCPD: circuit id 00000000
DHCPD: Allocate an address without class information (10.0.10.0)
DHCPD: subnet [10.0.10.1,10.0.10.254] in address pool main-pool is empty.
DHCPD: Sending notification of ASSIGNMENT FAILURE:
DHCPD: htype 1 chaddr cc04.0884.0000

DHCPD: remote id 020a00000a001e0200000000
DHCPD: circuit id 00000000
DHCPD: Sending notification of ASSIGNMENT_FAILURE:
DHCPD: due to: POOL EXHAUSTED
```

Podemos ver que, primeiramente, o roteador H1, (simulando uma PC) recebeu o IP 10.0.10.254 (subnet principal 10.0.10.0/24).

No roteador H1:

```
%DHCP-6-ADDRESS_ASSIGN: Interface Ethernet0/0 assigned DHCP address 10.0.10.254, mask
255.255.255.0, hostname h1
```

Mas, em seguida, o roteador H2 pede um endereço e o servidor responde dizendo que não existe endereço disponível (lembre-se que eu configurei apenas 1 endereço nessa rede 10.0.10.0/24)

Então, esse é o problema: **Como fazer para que o servidor de DHCP ofereça IPs na subnet secundária?**

ip dhcp smart-relay

http://www.cisco.com/en/US/docs/ios/12_3/ipaddr/command/reference/ip1_i1g.html#wp1081216

Esse comando faz com que o roteador passe a procurar IPs para endereços secundários, assim que recebe uma resposta dizendo que não há mais IPs disponíveis; como vimos nas mensagens acima.

Então, voltamos a simulação e acrescentamos esse comando nos dos roteadores ra e rb.

E, como mágica:

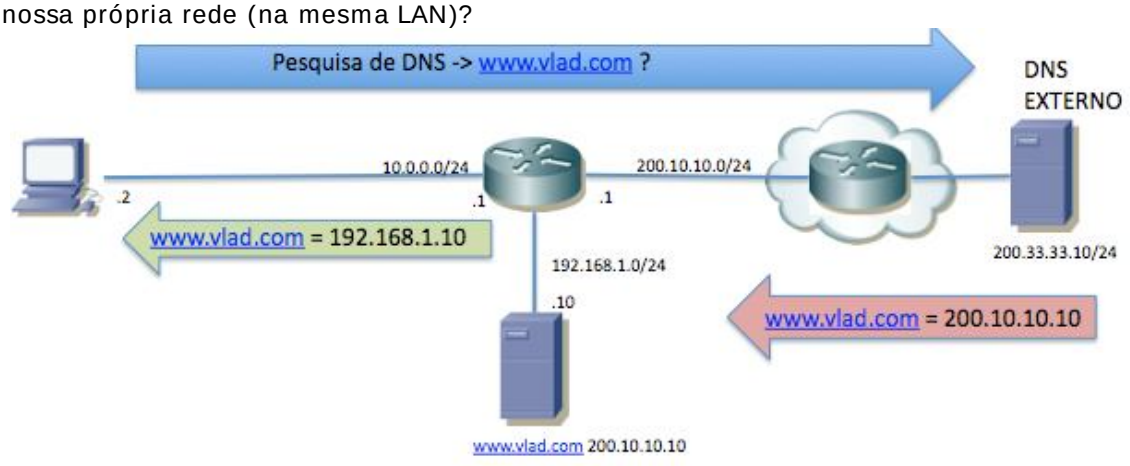
```
DHCPD: assigned IP address 10.0.20.11 to client 0063.6973.636f.2d63.6330.342e.3038.
3834.2e30.3030.302d.4574.302f.30.
DHCPD: Sending DHCPOFFER to client 0063.6973.636f.2d63.6330.342e.3038.
3834.2e30.3030.302d.4574.302f.30 (10.0.20.11).
DHCPD: unicasting BOOTREPLY for client cc04.0884.0000 to relay 10.0.20.4.
```

o roteador H2,

```
h2#
*Mar 1 02:06:05.935: %DHCP-6-ADDRESS_ASSIGN: Interface Ethernet0/0 assigned DHCP address
10.0.20.11, mask 255.255.255.0, hostname h2

h2#sh ip route
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
E1 - OSPF external type 1, E2 - OSPF external type 2
i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
ia - IS-IS inter area, * - candidate default, U - per-user static route
o - ODR, P - periodic downloaded static route

Gateway of last resort is 10.0.20.1 to network 0.0.0.0

10.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
S 10.0.30.6/32 [254/0] via 10.0.20.1, Ethernet0/0
C 10.0.20.0/24 is directly connected, Ethernet0/0
S* 0.0.0.0/0 [254/0] via 10.0.20.1
```

NAT - NATeando respostas de DNS em um roteador CISCO

A questão levantada foi:
Como usar um servidor de DNS externo para resolver nomes de um servidor web que esta dentro da nossa própria rede (na mesma LAN)?



Isso surgiu porque o servidor de DNS da empresa estava localizado num endereço publico (ex:

200.33.33.10) em uma outra cidade. E os computadores locais apontavam para esse servidor.

Mas o servidor de Web da empresa ficava na mesma LAN que estes computadores (em uma outra subnet ex: 192.168.1.10), mas era visível na Internet com o IP 200.10.10.10.

A dúvida então era, se a resposta do DNS vai ser 200.10.10.10 como as maquinas da rede 10.0.0.0/8 iriam alcançar esse endereço NATead (traduzido). Esse tipo de conexão, o roteador não aceita a conexão e "reseta" a mesma.

E podemos ver isto se tentarmos acessar o servidor de dentro da rede 10.0.0.0 usando o IP 200.10.10.10 como destino.

```
h1#telnet 200.10.10.10 80
Trying 200.10.10.10, 80 ...
% Connection refused by remote host
h1#
IP: tableid=0, s=10.10.10.2 (local), d=200.10.10.10 (Ethernet0/0), routed via FIB
IP: s=10.10.10.2 (local), d=200.10.10.10 (Ethernet0/0), len 44, sending
TCP src=38273, dst=80, seq=3597984194, ack=0, win=4128 SYN
IP: tableid=0, s=200.10.10.10 (Ethernet0/0), d=10.10.10.2 (Ethernet0/0), routed via RIB
IP: s=200.10.10.10 (Ethernet0/0), d=10.10.10.2 (Ethernet0/0), len 40, rcvd 3
TCP src=80, dst=38273, seq=0, ack=3597984195, win=0 ACK RST
```

Este TCP RST na verdade vem do roteador que esta realizando o NAT estático entre 192.168.1.10 e 200.10.10.10

```
nat#
IP: tableid=0, s=10.10.10.2 (Ethernet0/0), d=200.10.10.10 (Serial1/0), routed via RIB
NAT: [0] Allocated Port for 10.10.10.2 -> 200.10.10.1: wanted 27151 got 27151
NAT: i: tcp (10.10.10.2, 27151) -> (200.10.10.10, 80) [0]
NAT: s=10.10.10.2->200.10.10.1, d=200.10.10.10 [0]
IP: s=200.10.10.1 (Ethernet0/0), d=200.10.10.10, len 44, rcvd 6
TCP src=27151, dst=80, seq=1275777384, ack=0, win=4128 SYN
NAT: o: tcp (200.10.10.10, 80) -> (200.10.10.1, 27151) [24]
NAT: s=200.10.10.10, d=200.10.10.1->10.10.10.2 [24]
IP: tableid=0, s=200.10.10.10 (local), d=10.10.10.2 (Ethernet0/0), routed via FIB
IP: s=200.10.10.10 (local), d=10.10.10.2 (Ethernet0/0), len 40, sending
TCP src=80, dst=27151, seq=0, ack=1275777385, win=0 ACK RST
```

Assim seria necessário algum tipo de tradução automática dos dados enviados na reposta de DNS do servidor, para que as maquinas na rede 10.0.0.0/8 pudessem ir diretamente ao servidor web usando o endereço de destino 192.168.1.10.

E isso é possível através do que a CISCO chama de ALG (Application Level Gateway). Esse recurso permite ao IOS traduzir as respostas de DNS contidas dentro dos dados dos pacotes enviados pelo servidor. E não é necessária nenhuma configuração extra para que isso funcione. Mas existem restrições ao tipo de NAT usado!!

Quando estava trabalhando para aquela mesma empresa americana que eu citei no outro post, uma colega de trabalho que fazia o level 2 na época, me fez essa mesma pergunta: Como funciona o DNS quando existe um NAT envolvido?

Resolvi montar tudo num LAB dynamips e mostrar com debugs como isso funciona!

Tudo foi feito com roteadores (pc, servidor web, servidor dns inclusive).

As configurações do NAT são bem simples. Onde os endereços da LAN 10.0.0.0/8 são traduzidos para o endereço da interface externa 200.10.10.1. E existe um NAT estático traduzindo o endereço do servidor Web 192.168.1.10 para um endereço externo 200.10.10.10

```
nat#sh run | i Ethernet0/0|Serial1/0|nat
hostname nat
interface Ethernet0/0
```

```
ip nat inside
interface Serial1/0
ip nat outside
ip nat inside source list 10 interface Serial1/0 overload
ip nat inside source static 192.168.1.10 200.10.10.10
```

Uma conexão (ou ping) realizado de uma maquina externa sera feita diretamente com o IP 200.10.10.10

```
internet#ping www.vlad.com
Translating "www.vlad.com"...domain server (200.33.33.10) [OK]
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 200.10.10.10, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 160/194/248 ms
```

Realizando um teste de conexão de uma maquina na rede 10, podemos ver o resultado do ALG.

A conexão é feita com o endereço interno.

```
h1#telnet www.vlad.com 80
Translating "www.vlad.com"...domain server (200.33.33.10) [OK]
Trying www.vlad.com (192.168.1.10, 80)... Open
GET / HTTP/1.1
HTTP/1.1 400 Bad Request
Date: Mon, 01 Mar 1993 01:20:55 GMT
Server: cisco-IOS
Accept-Ranges: none
400 Bad Request
[Connection to www.vlad.com closed by foreign host]
```

No roteador de borda realizando o NAT podemos ver as traduções sendo realizadas

```
nat#
NAT: [0] Allocated Port for 10.10.10.2 -> 200.10.10.1: wanted 51523 got 51523
NAT: i: udp (10.10.10.2, 51523) -> (200.33.33.10, 53) [0]
NAT: s=10.10.10.2->200.10.10.1, d=200.33.33.10 [0]
NAT: o: udp (200.33.33.10, 53) -> (200.10.10.1, 51523) [46]
NAT: DNS resource record 200.10.10.10 -> 192.168.1.10
NAT: s=200.33.33.10, d=200.10.10.1->10.10.10.2 [46]
```

Podemos ver a pesquisa feita no servidor de DNS pelo endereço de origem (NATeado) 200.10.10.1

dns#
DNS: Incoming UDP query (id#49)
DNS: Type 1 DNS query (id#49) for host 'www.vlad.com' from 200.10.10.1(51523)
Send reply from internal information:
DOM: id=49, response, opcode=0, aa=0, tc=0, rd=1, ra=1
rcode=0, qdcount=1, ancount=1, nscount=0, arcount=0
query name is www.vlad.com, qtype=1, class=1
Answer section:
Name='www.vlad.com'
RR type=1, class=1, ttl=10, data length=4
IP=200.10.10.10
Authority section:
Additional record section:
DNS: Finished processing query (id#49) in 0.016 secs